

Name of Student: Pushkar Sane		
Roll Number: 45		Practical Number: 10
Title of Lab Assignment: To implement Pandas library in Python.		
DOP: 31-10-2023		DOS: 01-11-2023
CO Mapped: CO6	PO Mapped: PO3, PO5, PSO1, PSO2	Signature:

Practical No. 10

Aim: To implement Pandas library in Python.

1. Write a Pandas program to create and display a one dimensional array like object containing an array of data using the pandas module.
2. Write a pandas program to convert a dictionary to a Pandas Series.
3. Write a pandas program to create a dataframe from a dictionary and display it. sample data: {'X': [78,85,96,80,86], 'Y': [84,94,89,83,86], 'Z': [86,97,96,72,83]}
4. Write a pandas program to aggregate the two given data frames along rows and assign all data.
5. Write a pandas program to merge two given dataframes with different columns.

Description:

Creating and displaying a one-dimensional array-like object using the Pandas module involves the following key concepts:

1. Pandas Library:

Pandas is a powerful open-source data manipulation and analysis library for Python.

It provides data structures and functions for working with structured data, including Series and DataFrame.

2. Pandas Series:

A Pandas Series is a one-dimensional labeled array that can hold data of any data type.

It is similar to a NumPy array, but with the added feature of having an index associated with each element.

3. Creating a Pandas Series:

To create a Pandas Series, you can use the `pd.Series()` constructor, which accepts a variety of data types, including lists, dictionaries, and NumPy arrays.

The Series constructor allows you to specify the data and optionally customize the index.

4. Displaying a Pandas Series:

You can display the contents of a Pandas Series simply by using the `print()` function or by calling the Series object directly.

When you display a Pandas Series, you'll see both the data values and the associated index (default is an integer index).

Here's a breakdown of the steps in the Pandas program to create and display a one-dimensional array-like object:

1. Import the Pandas library using `import pandas as pd`.
2. Create an array of data (e.g., a list) that you want to convert into a Pandas Series. Use the `pd.Series()` constructor to convert the array of data into a Pandas Series.
3. Optionally, you can customize the index by providing it as an argument when creating the Series. Display the Pandas Series using the `print()` function or by calling the Series object directly.

Converting a dictionary to a Pandas Series using the `pd.Series()` constructor is a straightforward process. This allows you to leverage the powerful features of Pandas, such as data analysis, filtering, and manipulation, with your structured data. The resulting Series is a labeled one-dimensional data structure that can be accessed, indexed, and modified to suit your specific data analysis needs.

A Pandas DataFrame can hold data of various types, including integers, floating-point numbers, strings, or more complex objects.

Each column in the DataFrame can have its own data type, depending on the data provided.

In summary, creating a Pandas DataFrame from a dictionary is a powerful way to work with structured data in Python. It allows you to organize and manipulate data in a tabular format, making it easier to perform data analysis, filtering, and other operations on your data. The resulting DataFrame can be used for various data analysis tasks and is a fundamental building block in data science and data engineering.

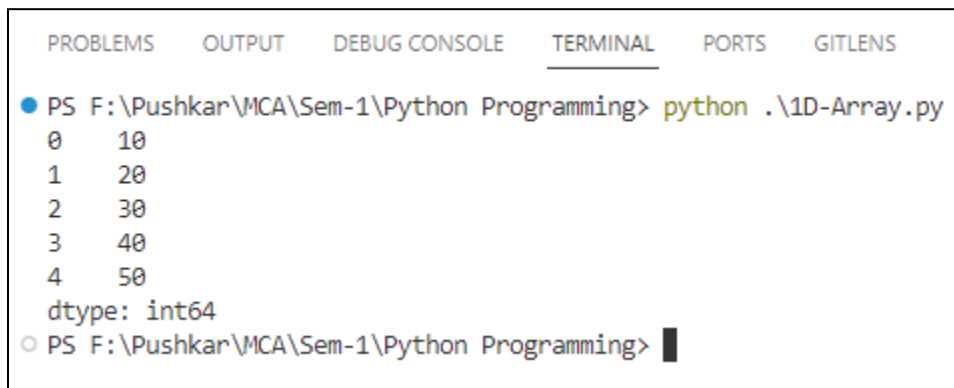
1. **Write a Pandas program to create and display a one dimensional array like object containing an array of data using the pandas module.**

Code:

```
import pandas as pd
data = [10, 20, 30, 40, 50]
my_series = pd.Series(data)
print(my_series)
```

Conclusion:

Here we have successfully created and displayed a one dimensional array-like object containing an array using pandas.

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS
PS F:\Pushkar\MCA\Sem-1\Python Programming> python .\1D-Array.py
0    10
1    20
2    30
3    40
4    50
dtype: int64
PS F:\Pushkar\MCA\Sem-1\Python Programming> █
```

2. **Write a pandas program to convert a dictionary to a Pandas Series.**

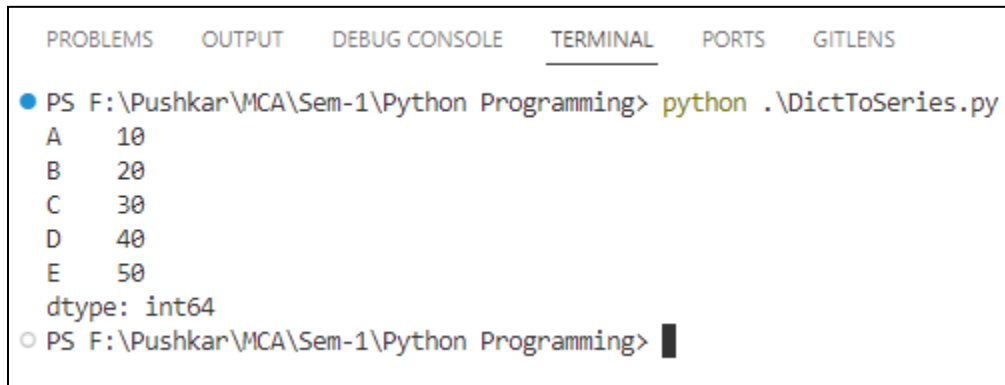
Code:

```
import pandas as pd
# Sample dictionary
data_dict = {'A': 10, 'B': 20, 'C': 30, 'D': 40, 'E': 50}
# Convert the dictionary to a Pandas Series
my_series = pd.Series(data_dict)

# Display the Pandas Series
print(my_series)
```

Conclusion:

Here we have successfully converted a dictionary to a Pandas Series.

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS
● PS F:\Pushkar\MCA\Sem-1\Python Programming> python .\DictToSeries.py
A    10
B    20
C    30
D    40
E    50
dtype: int64
○ PS F:\Pushkar\MCA\Sem-1\Python Programming> █
```

3. Write a pandas program to create a dataframe from a dictionary and display it.
sample data: {'X': [78,85,96,80,86], 'Y': [84,94,89,83,86], 'Z': [86,97,96,72,83]}.

Code:

```
import pandas as pd
```

```
# Sample data dictionary
```

```
data_dict = {'X': [78, 85, 96, 80, 86], 'Y': [84, 94, 89, 83, 86], 'Z': [86, 97, 96, 72, 83]}
```

```
# Create a Pandas DataFrame from the sample data
```

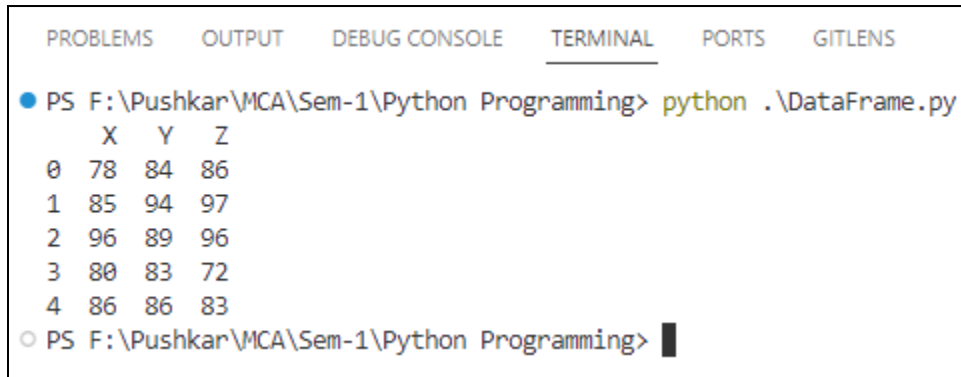
```
df = pd.DataFrame(data_dict)
```

```
# Display the DataFrame
```

```
print(df)
```

Conclusion:

Here we have successfully a dataframe from a dictionary for the given sample data.

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS
● PS F:\Pushkar\MCA\Sem-1\Python Programming> python .\DataFrame.py
      X   Y   Z
0  78  84  86
1  85  94  97
2  96  89  96
3  80  83  72
4  86  86  83
○ PS F:\Pushkar\MCA\Sem-1\Python Programming> █
```

4. Write a pandas program to aggregate the two given data frames along rows and assign all data.

Code:

```
import pandas as pd
```

```
# Sample data for two DataFrames
```

```
data1 = {'A': [1, 2, 3], 'B': [4, 5, 6]}
```

```
data2 = {'A': [7, 8, 9], 'B': [10, 11, 12]}
```

```
# Create the first DataFrame
```

```
df1 = pd.DataFrame(data1)
```

```
# Create the second DataFrame
```

```
df2 = pd.DataFrame(data2)
```

```
# Concatenate the two DataFrames along rows and assign all data
```

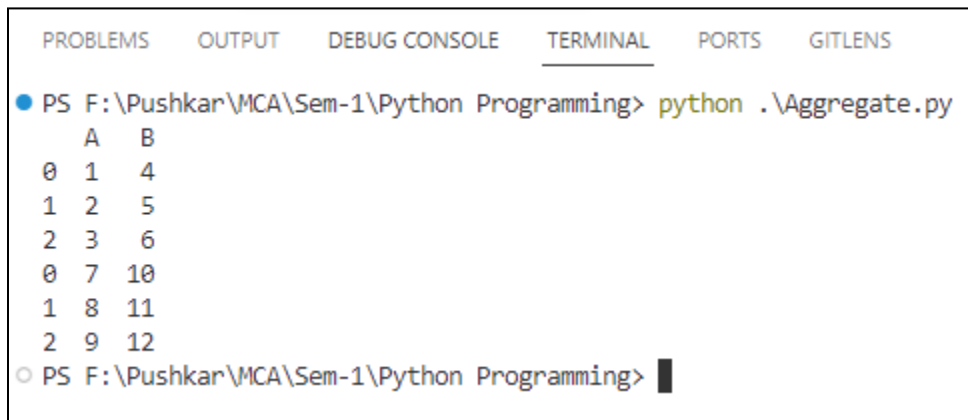
```
result = pd.concat([df1, df2])
```

```
# Display the aggregated DataFrame
```

```
print(result)
```

Conclusion:

Here we have successfully aggregated the two given data frames along rows and assigned all data.

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS
PS F:\Pushkar\MCA\Sem-1\Python Programming> python .\Aggregate.py
  A  B
0  1  4
1  2  5
2  3  6
0  7 10
1  8 11
2  9 12
PS F:\Pushkar\MCA\Sem-1\Python Programming>
```

5. Write a pandas program to merge two given dataframes with different columns.**Code:**

```
import pandas as pd
# Sample data for two DataFrames
data1 = {'ID': [1, 2, 3], 'Name': ['Prasad', 'Anish', 'Shreya']}
data2 = {'ID': [2, 3, 4], 'Age': [25, 30, 35]}

# Create the first DataFrame
df1 = pd.DataFrame(data1)

# Create the second DataFrame
df2 = pd.DataFrame(data2)

# Merge the two DataFrames on the 'ID' column
merged_df = pd.merge(df1, df2, on='ID', how='inner')

# Display the merged DataFrame
print(merged_df)
```

Conclusion:

Here we have successfully merged two given dataframes with different columns.

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS

● PS F:\Pushkar\MCA\Sem-1\Python Programming> python .\MergeDataFrame.py
  ID  Name  Age
0   2  Anish  25
1   3  Shreya  30
○ PS F:\Pushkar\MCA\Sem-1\Python Programming> █
```